Computers and Visualization in Mathematics Education: Some Thoughts

Ronald D. Notestine

The Role of Computers in Mathematics Education

> *"Some mathematician, I believe, has said that true pleasure lies not in the discovery of truth, But in the search for it."*
> *··· Tolstoy, Anna Karenina*

It is a cliche' to say that we are now in the beginning stages of a revolution in education. But, it is true nonetheless. Computers can be used to create a learning environment for the student in which the student can be guided to do real mathematics on his or her own: An environment, in which there are the important elements of surprise and discovery.

Effective learning requires that the learner be personally involved. This statement is a tenet of the educational theories of Piaget, Dewey, and Montossori. (Abelson and DiSessa 1980 ; Papert 1980) Despite this, mathematics education in both America and Japan consists almost entirely of a collection of sample problems and techniques for their solution. That is a collection of algorithms, with the student to memorize them and learn how to recognize which algorithm should be applied to a given problem. The algorithms themselves are rarely very interesting to the average student, being usually of very old vintage themselves. Almost never does a student have the chance to do and discover mathematics on his or her own, "···to approach mathematics by doing it rather than only learning about it···" (Abelson and DiSessa 1980)

The dangers of teaching only specific algorithms are well known. They normally work

only on limited problems specially prepared for use in the classroom or on a test. Once the test for which an algorithm was memorized is past, the algorithm is soon forgotten. Most students do not find these (usually ancient) algorithms very interesting. In the real world, problems are rarely tailor made to fit classroom algorithms. (Note : In comparing students who are taught specific techniques to solve expected test problems, and students who are taught concepts, both groups normally get the same scores on tests, *if the questions are what the former group* expected. But, when questions are conceptually different from what the student expects, the former group normally have great difficulty and score much lower than the latter. (Tall 1991))

Very rarely does a student ever get a chance to do mathematics as it is actually done in engineering, science, or other applied fields, such as economics. This is in part because "real-world" problems usually involve computational and modeling complexities. Complexities great enough to normally exclude them from use in the normal classroom.

It is ironic that mathematicians have been slower than writers and poets to adopt the computer. This is perhaps due in part to what most mathematicians actually do : They reason with symbols and pictures. Unlike what many think, most mathematicians rarely calculate with actual numbers. Actual mathematical calculations are far more often done by scientists, engineers, economists, and those in business. In the early days computers were used almost exclusively for numerical calculations. Thus, most mathematicians had little use for them. Later, word processors became common. But, they still could not easily handle mathematical notation. So, even though the poets and writers started "joining the revolution" at this point, mathematicians still stayed away. No small part of the problem was probably a natural conservativeness among members of the field. There seemed to be a wide consensus, with Einstein, that the proper tools of the pure researcher were paper, pencil, and "perspiration". (Clark 1971)

The advent of Computer Aided Design systems and computer graphics made some change in mathematics education. However, there was no real change in the situation among mathematicians. While a graphic artist could use a CAD system or drawing program to make more illustrations in less time for mathematics textbooks, each illustration has to be produced separately by someone specially trained in the use of the program. While such programs found great use among architects and engineers, there was no way to "hook the picture to the underlying mathematics" and observe changes.

The advent of symbolic algebra systems with graphical capability now allow us to do mathematics "hooked" to a picture.

Computer Algebra Systems

> *Students simply do not retain for long what they learn by imitation from lectures, worksheets, or routine homework. Presentation and repetition help students do well on standardized tests and lower-order skills, but they are generally ineffective as teaching strategies for long-term learning for higher-order thinking, and for versatile problem-solving.*
>
> National Research Council in its report:
>
> *Everybody Counts: A Report to the Nation About the Future of Mathematics Education*

The impact of Computer algebra and graphing systems is not just in their ability to aid in making concepts visual. They have the potential to involve a much wider population of students in actually doing mathematics, symbolic and numeric, as well as graphic.

Not all computer mathematics, or symbolic algebra packages are graphical. A large number are purely symbolic. A report prepared by the Committee on the Undergraduate Program in Mathematics of the Mathematical Association of America, "Priming the Calculus Pump: Resources and Innovations", lists ten pages of software titles for doing mathematics. (Tucker 1991) The titles are divided among sixteen different categories, and so include some multiple listings for multi-purpose programs. However, even allowing for that, there would seem to easily be over 200 different titles of software for doing or teaching some aspect of mathematics.

Regardless of whether the program is visual, symbolic, or both, the computer can cause the student to be actively involved. A good example of a widely used, but non-visual, program is ISETL (Interactive SET Language). Originally devised on main frame computers as the set language SETL, it has been moved to the personal computer, made interactive, and is used in undergraduate education. (Not the least reason that it is used so widely is that the authors distribute it free of charge. (Dubinsky 1992)) In the preface to their textbook, "Learning discrete Mathematics using ISETL", Baxter et.al. write: " ···what the student does in order to learn is much more important than what the professor does in order

to teach. ··· ISETL is a programming language ··· our intention is for students to write code, think about what they have written, predict its results, and run their programs to check their predictions. ··· A powerful effect of computers for most students is to keep them deeply involved. This can be very important, because we believe that any time you construct something on the computer then, with a lesser or greater degree of awareness, you will also construct something in your head." (Baxter and others 1989)

Mathematical Visualization

> " ··· *intuitive understanding fosters a more immediate grasp of the objects one studies, a live rapport with them, so to speak, which stresses the concrete meaning of their relations.*"
>
> David Hilbert in his preface to *Geometry and the Imagination*

General

There has always been an important place for visual information and diagrams in mathematics. The ancient Greeks made extensive use of figures drawn in the sand. In fact, to make visible is the original meaning of δειχνυμι (deiknumi, ancient Greek meaning "to prove"). (Klotz 1991) Descartes and Newton were still working in what might be called a "visual mode". Newton's Principia was formulated entirely in geometric, diagrammatic arguments. Our language still uses visual metaphors for understanding: "Can you see that ···", "Can you show me ···".

In the introduction *What is Mathematical Visualization* to the Mathematical Association of America volume *Visualization in Teaching and Learning Mathematics*, the editors say "We take the term *visualization* to describe the process of producing or using geometrical or graphical representations of mathematical concepts, principles or problems, whether hand drawn of computer generated." (Zimmerman and Cunnningham 1991)

The United States Board on Mathematical Sciences of the National Research Council states in a 1990 report: "In recent years computer graphics have played an increasingly important role in both core and applied mathematics, and the opportunities for utilization are enormous."(NRC 1990)

A report to the United States National Science Foundation (McCormick 1987) states:

"Visualization ··· transforms the symbolic into the geometric, enabling [users] to observe their simulations and computations. Visualization offers a method of seeing the unseen. It enriches the process of ··· discovery and fosters profound and unexpected insights." (McCormick and others 1987) The report was specifically speaking of visualization in the applied sciences, but the results apply equally well to mathematical applications in any number of fields, including economics and management.

In a 1979 paper, Davis and Anderson (Davis and Anderson 1979) say, "Mathematics has elements that are spatial, kinesthetic, elements that are arithmetic or algebraic, elements that are verbal, programmatic. It has elements that are logical, didactic and elements that are intuitive, or even counter-intuitive ··· These may be compared to different modes of consciousness. To place undue emphasis on one element or group upsets a balance. It results in an impoverishment of the science and represents an unfulfilled potential. ···We must not block off any mode of experience or thought." They were making a strong plea to reverse the trend of the last hundred years, or so, in mathematics, which had cut mathematics from its modern roots. They pleaded "Restore geometry. Restore experimental and intuitive mathematics. Give a proper place to computing and programmatics. Make full use of computer graphics". The call for the use of more computer graphics in mathematics was visionary for its time. That is, visionary given the state of relative infancy of CAD systems, requiring mini or mainframes, and primitiveness of desktop computer graphics at that time.

In an article in the New York Academy of Sciences magazine, The Sciences, titled "Picture Puzzling", Ian Rival tells us, "Diagrams are, of course as old as mathematics itself. Geometry has always relied heavily on pictures, and, for a time, other branches of mathematics did too. Even Isaac Newton ··· did not actually prove [the] fundamental theorems. ···Had you asked him to justify them, he would likely have presented an argument that, though compelling, was loose and depended heavily on pictures. ···The intuitive and often pervasive style of argument used by Newton and his contemporaries fell into disrepute in the nineteenth century after it proved, in several celebrated cases to be misleading. ···The limits of deductivism are at last dawning on mathematicians, thanks largely to computers." (Rival 1987)

As Rival says, there has been a move back toward the use of the visual, which has been largely aided by the capabilities of the computer. This is supported by Steen in his article in Science, "The Science of Patterns". Steen writes "Mathematics is often defined as the

science of space and number, as the discipline rooted in geometry and arithmetic. Although the diversity of modern mathematics has always exceeded this definition, it was not until the recent resonance of computers and mathematics that a more apt definition became fully apparent. mathematics is the science of patterns. The mathematician seeks patterns in number, in space, in science, in computers and in imagination. Mathematical theories explain the relations among patterns; functions and maps, operators and morphisms bind one type of pattern to another to yield lasting mathematical structures. Applications of mathematics use these patterns to 'explain' and predict natural phenomena that fit the patterns. Patterns suggest other patterns, often yielding patterns of patterns." (Steen 1988) In commenting on this passage, Zimmerman et.al. note : "In speaking of 'patterns', Steen is speaking mataphorically, but the metaphor of patterns is surely a visual metaphor. Not all patterns can be visualized, but it is as natural to want to visualize a pattern as it is to want to hear a melody. If mathematics is the science of patterns, it is natural to try to find the most effective ways to visualize these patterns and to learn to use visualization creatively as a tool for understanding. This is the essence of mathematical visualization." (Zimmerman and Cunnningham 1991)

G. H. Hardy in his famous essay "A Mathematicians Apology", begins by comparing mathematicians to painters and poets as makers of patterns. (Hardy 1956) Hardy makes it clear that he means patterns of ideas, rather than the painter's colors, or the poet's words. But, still, one wonders what hardy might have said about modern computer graphic and symbolic algebra systems.

Henri Poincaré (1854-1912) was renowned as the greatest all-round mathematician of his time. He is considered the last person to comprehend the entire body of known mathematics. He was also a mathematical physicist, who was nibbling at the edges of special relativity when Einstein published his 1905 paper.(Clark 1971) He was a noted writer and popularizer who won the French gran prix for literature for his expository science writings.

By way of founding both modern topology and dynamical systems theory during the last half of the mineteenth century, he is also the founder of modern chaos theory.(Gleick 1987) Of particular interest are his writings on the creation of mathematics. (Poincaré 1956) He refers to mathematics as a logical series of syllogisms. This is interesting for one who founded so visually oriented a field as topology, not to mention dynamical systems. He would seem to have had an excellent visual intuition. But, also, Einstein is known to have been extremely visual in his thinking. (Clark 1971) His understanding usually being in very

concrete images. Was that why it was Einstein who first formulated special relativity? That is, Einstein had the stronger visual understanding, and hence could better see how to bring it all together. In any case, we know that Poincaré understood the existence of chaos in dynamical systems, but, after standing on the shore and gazing out, so to speak, left for lack a way to explore this new ocean. The computer, and it visual capabilities, have brought us back to Poincaré's shore, and shown us spectacular vistas.

Coffey et.al., in their 1990 paper in Science, write "Several developments underlie the present revival in classical mechanics, including computerized algebraic processors and color graphics. ··· Color graphics prove invaluable in visualizing the global behavior and discovering minutiae of local behavior hidden beneath the mass of calculation. Pseudocoloring a function over a domain, a widespread technique in applied mathematics, has produced stunning pictures; they have opened the eyes of mathematicians to hitherto unsuspected phenomena in the dynamics of nonlinear maps." (Coffey and others 1990)

Visualization in Mathematics Education

We are concerned with visualization and its importance in regard to education. When we use the term visualization form this point on, we will normally mean a computer-based approach to visualization.

Above, we discussed visualization in the sciences. This has become so widespread that the term "scientific visualization" has become a buzzword. But it is still an important term as it represents serious and fruitful new approaches to doing science. The report "Visualization in Scientific Computing", issued by the Unites States National Science Foundation, describes it as, not computer graphics, but something that uses the technology of computer graphics to visually represent things otherwise difficult or impossible to see or humanly comprehend.

In education, visualization can be used to assist and speed understanding in students who otherwise have difficulty. It can also be used to deepen the nature of the understanding. Now, the mathematics studied by students almost exclusively involves artificially simple problems in constrained and static circumstances. The reasons for this are undoubtedly many. But, they certainly include the time and difficulty of the calculations involved in realistic problems. and the representation of the resulting information, the answer.

Three Types of Visualization

Primarily drawn from the work of Mike Keeler of Stardent Computers, summarized by Cunningham in "The Visualization Environment For Mathematics Education." (Cunningham 1991)

□ Postprocessing

In Postprocessing the knowledge is complete, and the user creates a display of the finished product.

This is essentially no different from the traditional blackboard drawing or book illustration that we have always used.

The only real difference possibly lying in the fact that the user does something to cause the computer to produce the picture. If the user must thoughtfully construct a command, the effect of conscious involvement might cause more attention to be paid and make this more effective than a book illustration. If there is little or no thought needed to cause the picture to be produced, there may be little difference between this and a book illustration. One important difference must surely be the flexibility of even a postprocessing system: The student cannot change the illustration in a book, but can can modify the visual produced by a computer program.

□ Tracking

The knowledge is developed and the student watches it as it is progressively displayed to see its nature.

This is essentially Postprocessing, where the illustration is now a movie.

□ Steering

The user is in the processing loop and can interact with and manipulate the simulation as it is underway.

□ Summary

"Educational visualization can take advantage of these ideas. A simulation can present static images that report the results of a single computation, dynamic images that illustrate the behavior of a sequence of computations, or a steerable behavior that let's students

manipulate the simulation as it progresses.

It is an old saw that, when we first try to use something new, we tend to use it in familiar ways. Computer graphic capability is no exception. The great majority of educational programs that the author has seen on the market in Japan over the last ten years has resembled nothing so much as the pages of a book of drill exercises. (In most cases, the graphics actually illustrated nothing educational at all, but were merely for 'prettifying' effect.) Most illustrative use of computer graphics anywhere so far has been static, what is called postprocessing, above.

Technology

The machines used for scientific visualization range from super computers to desktop personal computer. In education, we clearly do not have the money for many, if any, super computers, and must be happy with some small number of desktop stations. This means that any software written for educational purposes must meet this lowest common denominator test: It must run on a low cost Macintosh, a low cost IBM PC or compatible system, or (in Japan) on a standard model of the NEC 9800 series. A recent article (Cunningham, 1990), also lists desirable capabilities as: 1) Easy and flexible user interaction, 2) High resolution graphics, 3) Dynamic and animated displays, 4) Quick response to recomputation, and 5) Multitasking so students could do other work during a long computation.

The *Mathematica* environment certainly fulfills all of these requirements, with the exception of "easy". To display sophisticated visual information can range from quite easy to complicated. In general, requiring some knowledge of the programming language. On the other hand, *Mathematica* offers a complete environment for doing mathematics, and the programming language is a complete programming environment in its own right. So, the investment in learning the necessary skills to use the package pays its own substantial dividends.

Mathematica Graphics an Overview

There are two parts to *Mathematica* : the Kernel, which performs all calculations, and the Front End, which is the user interface. The kernel is what does all the mathematics, and may be thought of as the *Mathematica* program proper. The front end handles all interaction with the user. It receives information from the user and passes it on to the kernel. It receives results from the kernel, and formats them for human consumption. *Mathematica* runs on a wide variety of computer systems, from PC's to large mini's. In all cases *the kernel operates identically*. It is only the front end which is different for different systems. Also, the front end is considerably smaller than the kernel. By dividing the labor in this way, relatively inexpensive machines can be used to run the front end and communicate, via a network, with a faster, more expensive machine running a *Mathematica* kernel. Because, the kernel always operates in the same way, any front end can communicate with a kernal running on any machine. This provides complete flexibility in mixing machines on a network. It also means that, even though some computers have a more convenient interface, they all have the same actual *Mathematica* commands and syntax. If a user is sitting at a Macintosh which is connected on a high-speed network with a DEC Vax running a *Mathematica* kernel, the user will see what appears to be a purely Macintosh-interface work session. The only way the user would know that the kernel is not running on the Macintosh would be speed. The kernel on the Vax would run much faster than one on any Macinotosh, of course.

The Macintosh supports the Notebook Front End. Notebooks are sophisticated, interactive electronic documents. They mix computations, text, graphics, and animations. In the latest version of *Mathematica*, they support sound and moving video clips as well. On the computer screen, units of text, graphics etc can be freely opened and closed for convenient viewing.

What can be seen can also be easily printed. This paper is being made as a *Mathematica* notebook on a Macintosh computer. While motion and sound cannot be transferred to the printed page, just about anything else can. In the case of motion and sound, it is noteworthy that *Mathematica* notebooks are becoming a standard of commnication at many universities and research institutions.

From now, any mention of the *Mathematica* front end will mean the Macintosh Notebook Front End.

The "units" mentioned above are called "cells" . The notebook front end provides for a rather dazzling variety of cells for different purposes. For example, this sentence, and (almost) all of its predecessors, resides in a "text" cell. The heading in large type reading "*Mathematica* Graphics an Overview" is in a type of cell called "Section Heading."

However, there are only two types of cell that have to do with the kernel, with *Mathematica* proper : "Input" cells, and "Output" cells.

-Input cells contain information to send to the kernel for calculation or processing. Each input cell is numbered sequentially from one. This number does not change during a *Mathematica* session.

-Output cells always correspond to an input cell and contain a result. An output cell is always numbered with the same number as the input cell that produced it. (Output cells are not always displayed. There may be no particualr result to display, or the user may have specified that that output not be displayed.)

Following is an example of a *Mathematica* session. Other than a few preliminaries, only graphics will be discussed. There will be no mention of the extremely powerful symbolic rule and pattern matching capabilites of *Mathematica*. Nor any discussion of equation solving, symbolic and numeric differentiation and integration, differential equation solving, nor a host of other capabilities. Nor will there be any discussion of *Mathematica*'s extensive programming language, and capabilitues for linking to external programs and files.

The disussion will cover some of the basics of graphics in *Mathematica*. Even then, a great deal is left out. For example, animation capabilites are not shown. The printed page cannot, of course, show an animation. However, student texts using *Mathematica* now routinely include electronic notebooks, and these can include animations for students to view, or modify for themselves.

Some Preliminaries

□ Basic Operations

Arithmetic operations are evauated and the result Returned in the output cell

```
In[1]:=
    3+4
Out[1]=
    7
```

There are over a thousand built-in functions. "Pi" is taken to be the standard "π". Notice that *Mathematica* leaves the result in exact form, rather than converting to approximate, numeric form.

```
In[2]:=
  Sin[Pi/4]

Out[2]=
      1
  ----------
  Sqrt[2]
```

Notice that square brackets, [···] , are used to delimit function arguments.

We can force *Mathematica* to evaluate an expression to numeric form using the built-in function "N". The percent sign (%) refers to the previous ouptut. So, 1/Sqrt [2] is evaluated to 0.707107.

```
In[3]:=
  N[%]

Out[3]=
  0.707107
```

Mathematica normally shows only six places of precision. However, by default, all calculations are carried out to 19 places of precision. The user can specify any arbitrary nunber of places of precision. (See the calculation of 100 places of π, below.) The full precision is retained in memory at all times, regardless of whether a lesser number of places is displayed. The percent sign can be used to refer to other ouputs Here is the square of the first output.

```
In[4]:=
  %1^2

Out[4]=
  49
```

Mathematica will always attempt to retain the rational form of an expression. (Notice that a space between symbols means multiplication)

```
In[5]:=
  25 4 / 15

Out[5]=
  20
  --
  3
```

Again, we can use the numeric function, N, to force evaluation.

```
In[6]:=
  N[%]
Out[6]=
  6.66667
```

Mathematica can perform calculations to arbitary precision. So to find 100 places of Pi ···

```
In[7]:=
  N[Pi,100]
Out[7]=
  3.14159265358979323846264338327950288419716939937510\
    58209749445923078164062862089986280348253421170\
    68
```

Mathematica can do these calculations with great speed. The first one hundred places of Pi, above, was returned in a small fraction of a second We can check this by asking for the Timing. This will return the amount of CPU time the kernel devoted to the calculation. It will often underestimate by some amount, as the kernel has no way of knowing how much time the front end requires to render the result. For a complicated graphic, this can sometimes be a significant amount of time.

For 900 places of Pi, on a 25 MHZ Macintosh Powerbook 170, the calculation takes about 2.6 seconds of Kernel time.

```
In[8]:=
  N[Pi, 900]  //Timing
Out[8]=
  {2.56667 Second, 3.14159265358979323846264338327950288\
    41971693993751058209749445923078164062862089986280\
    34825342117067982148086513282306647093844609550582\
    23172535940812848111745028410270193852110555964462\
    29489549303819644288109756659334461284756482337867\
    83165271201909145648566923460348610454326648213393\
    60726024914127372458700660631558817488152092096282\
    92540917153643678925903600113305305488204665213841\
    46951941511609433057270365759591953092186117381932\
    61179310511854807446237996274956735188575272489122\
    79381830119491298336733624406566430860213949463952\
    24737190702179860943702770539217176293176752384674\
    81846766940513200056812714526356082778577134275778\
    96091736371787214684409012249534301465495853710507\
    92279689258923542019956112129021960864034418159813\
    62977477130996051870721134999999837297804995105973\
    17328160963185950244594553469083026425223082533446\
    85035261931188171010003137838752886587533208381420\
    617177669147304}
```

□ Expressions in Mathematica

Some of the types of expression in *Mathematica* are:

- Numbers
- Symbols
- Strings
- Compound Expressions

We can assign a value to the symbol x

```
In[9]:=
  x = 12
Out[9]=
  12
```

If we enter x, it evaluates to 12.

```
In[10]:=
  x
Out[10]=
  12
```

We can peform operations with the symbol

```
In[11]:=
  x - 3
Out[11]=
  9
```

But, the symbol "y" has no value, so ···

```
In[12]:=
  y-3
Out[12]=
  -3 + y
```

The result is just -3 plus whatever "y" is (as yet unknown).

Double quotes are used to define strings. (The quotes are omitted from the output.)

```
In[13]:=
  "This is a string."
Out[13]=
  This is a string.
```

The function "InputForm" shows us the input needed to produce an output.

```
In[14]:=
   InputForm[%]

Out[14]//InputForm=
   "This is a string."
```

Mathematica groups objects with lists. Curly brackets are used to delimit lists.

```
In[15]:=
   shortList = {1,2,a,b,x}

Out[15]=
   {1, 2, a, b, 12}
```

Notice that "x" still has the value 12.

About UPPER and lower case letters. *Mathematica* distinguishes between cases. So, the word "List" is not the same as "list". "List", with an upper case "L", is a reserved word, and could not be used to name an expression.

When using *Mathematica* functions, care must be taken with capitalization.

As an example:

```
In[16]:=
   Sin[Pi/2]

Out[16]=
   1
```

But,

```
In[17]:=
   sin[Pi/2]

   General::spell1:
      Possible spelling error: new symbol name "sin"
        is similar to existing symbol "Sin".

Out[17]=
   sin[Pi/2]
```

Mathematica does not know what "sin", with a lower case "s" is, so it just returns what it reveived, unchanged. However, if you are using version 2.0, or later, there is a spelling checker that detects when a symbol resembles an existing *Mathematica* symbol. In the case above, if you are using version 2.0 or later, you would get the message:

```
General::spell1:
   Possible spelling error: new symbol name "sin"
     is similar to existing symbol "Sin".
```

Returning to the exposition ···

Another way, perhaps the most sommon, to generate lists is to use the **Table** command.

Table has several forms. To repeat an expression 5 times

In[18]:=

```
Table[k,{5}]
```

Out[18]=

```
{k, k, k, k, k}
```

To go from 1 to 5 in steps of one

In[19]:=

```
Table[k,{k,5}]
```

Out[19]=

```
{1, 2, 3, 4, 5}
```

To make a list of lists

In[20]:=

```
Table[{k,k^2},{k,5}]
```

Out[20]=

```
{{1, 1}, {2, 4}, {3, 9}, {4, 16}, {5, 25}}
```

We can use the command TableForm to put this into a more readable form

In[21]:=

```
TableForm[%]
```

Out[21]//TableForm=

```
1   1
2   4
3   9
4   16
5   25
```

To see how *Mathematica* actually stores an expression, we can use the function **FullForm**.

In[22]:=

```
FullForm[shortList]
```

Out[22]//FullForm=

```
List[1, 2, a, b, 12]
```

This tells us that the expression named "shortList" is a list, and its elements are '1', '2', 'a', 'b', and '12'.

The Headof "shortList" is List.

In[23]:=

```
Head[shortList]
```

Out[23]=

```
List
```

This tells us what sort of object shortList is.

We can find the length of it using the Length function

In[24]:=

```
Length[shortList]
```

Out[24]=

```
5
```

We can separate a part of the shortList using the Part function

In[25]:=

```
Part[shortList,4]
```

Out[25]=

```
b
```

Double square brackets are used to denote a part of an expression

In[26]:=

```
shortList[[4]]
```

Out[26]=

```
b
```

□ Options in *Mathematica*

Options are used to alter a large number of *Mathematica* commands. All graphics commands have a wide variety of options. The Options command can be used to see the options available for a perticular command.

To see the options for the Plot command

```
In[27]:=
  Options[ Plot ]
Out[27]=
  {AspectRatio -> 1/GoldenRatio, Axes -> Automatic,
    AxesLabel -> None, AxesOrigin -> Automatic,
    AxesStyle -> Automatic, Background -> Automatic,
    ColorOutput -> Automatic, Compiled -> True,
    DefaultColor -> Automatic, Epilog -> {},
    Frame -> False, FrameLabel -> None,
    FrameStyle -> Automatic, FrameTicks -> Automatic,
    GridLines -> None, MaxBend -> 10.,
    PlotDivision -> 20., PlotLabel -> None,
    PlotPoints -> 25, PlotRange -> Automatic,
    PlotRegion -> Automatic, PlotStyle -> Automatic,
    Prolog -> {}, RotateLabel -> True,
    Ticks -> Automatic, DefaultFont :> $DefaultFont,
    DisplayFunction :> $DisplayFunction}
```

The list returned shows the options with the default value for each option.

For example, the default value for the AspectRatio, which is the ratio of scales for the horizontal and vertical axes, is the reciprocal of the golden ratio. (The golden ratio is 1.618 ···, and its reciprocal is 0.618···)

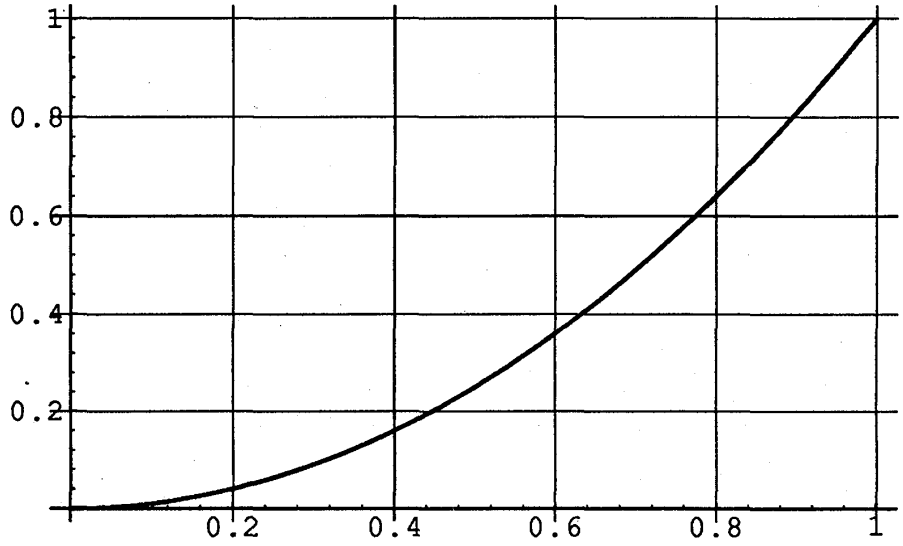
To see what the default is for any option, for example GridLines,

```
In[28]:=
  Options[Plot, GridLines]
Out[28]=
  {GridLines -> None}
```

To change the grid lines option to automatic when making a plot

In[29]:=

```
Plot[ x^2, {x,0,1}, GridLines->Automatic ]
```



Out[29]=

```
-Graphics-
```

The default value for an option can be changed by using the SetOptions command

In[30]:=

```
SetOptions[Plot, GridLines->Automatic];
```

(Notice the semicolon to supress output. Otherwise the entire list of Plot options would be returned.)

Now, if we ask what the default setting for the GridLines option is

In[31]:=

```
Options[Plot, GridLines]
```

Out[31]=

```
{GridLines -> Automatic}
```

Resetting the default to its orignal value ···

In[32]:=

```
SetOptions[Plot, GridLines->None];
```

Graphics Commands

□ Function Plotting

Mathematica has a large number of built-in commands for plotting functions.

(By 'function', is meant a *Mathematica* expression which evaluates to a real number or a list of real numbers)

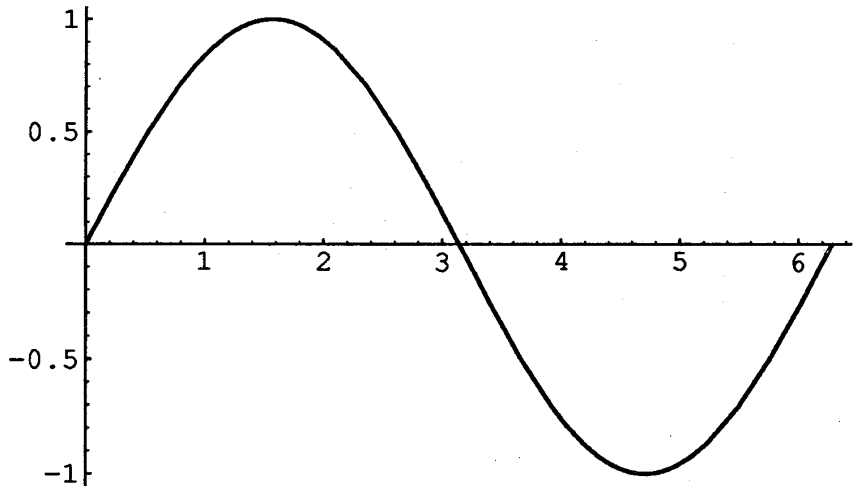
These are *Mathematica* built-in functions which produce a graph or chart

Plot
Plot3D
DensityPlot
ParamtricPlot
ContourPlot
ParametricPlot3D
Play

The classic first graph is the sine function from 0 to 2 Pi

In[33]:=

```
Plot[ Sin[x], {x,0,2Pi}]
```



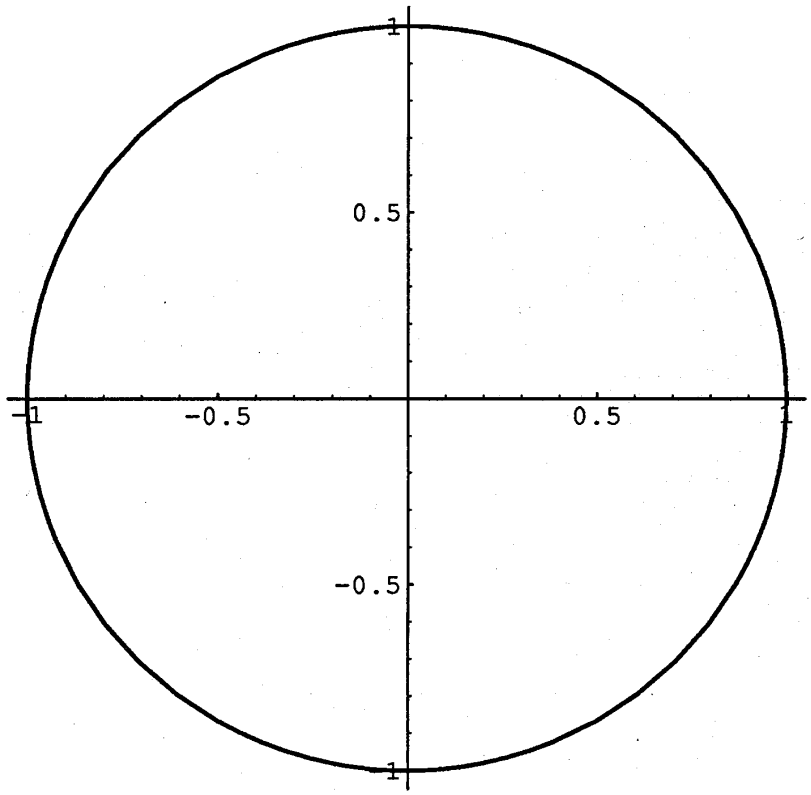
Out[33]=

```
-Graphics-
```

If the x and y values both depend on time, we can use a parametric plot. The simplest example is a circle, x = cos(t) and y = sin(t).

In[34]:=

```
ParametricPlot[{Cos[t],Sin[t]},{t,0,2Pi},
                AspectRatio->1]
```



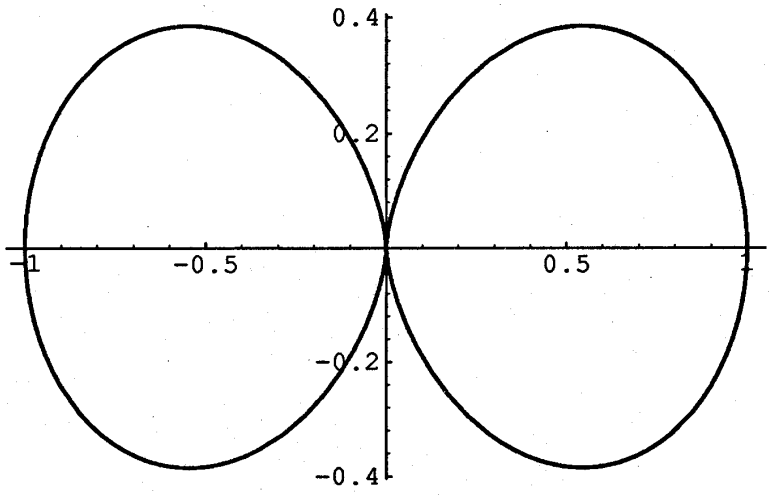
Out[34]=

```
-Graphics-
```

A more sophisticated example might be $x = \cos^3(t)$ and $y = \cos^2(t)\sin(t)$

In[35]:=

```
ParametricPlot[{Cos[t]^3, Cos[t]^2 Sin[t]},
               {t,0,2Pi}]
```



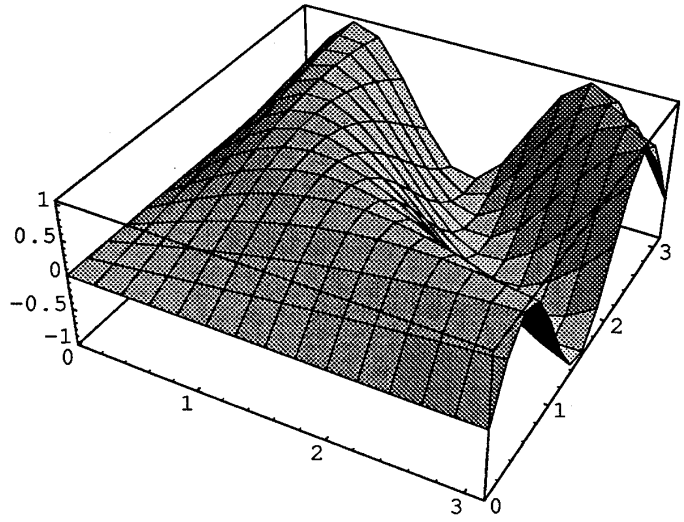
Out[35]=

```
-Graphics-
```

We can plot 3-dimensional surfaces. Here is one where the height (z direction) is given by $z = \sin(xy)$. (That is the sine of x times y.)

In[36]:=

```
Plot3D[ Sin[x y],{x,0,Pi},{y,0,Pi}]
```



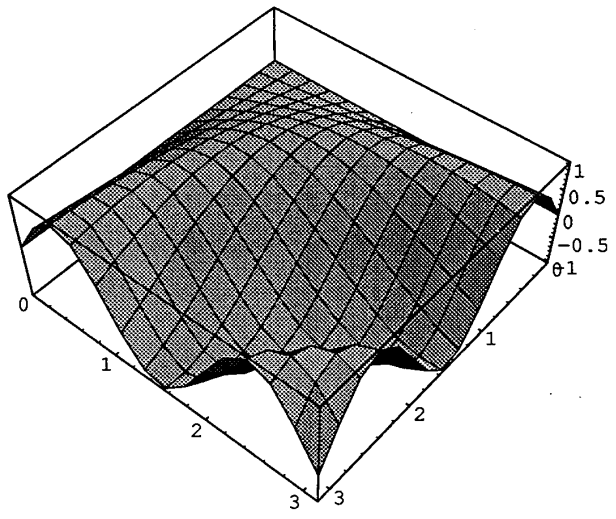
Out[36]=

```
-SurfaceGraphics-
```

We can change the viewpoint of a three dimensional plot.

In[37]:=

```
Plot3D[ Sin[x y],{x,0,Pi},{y,0,Pi},
        ViewPoint->{1.734, 1.477, 2.502}]
```



Out[37]=

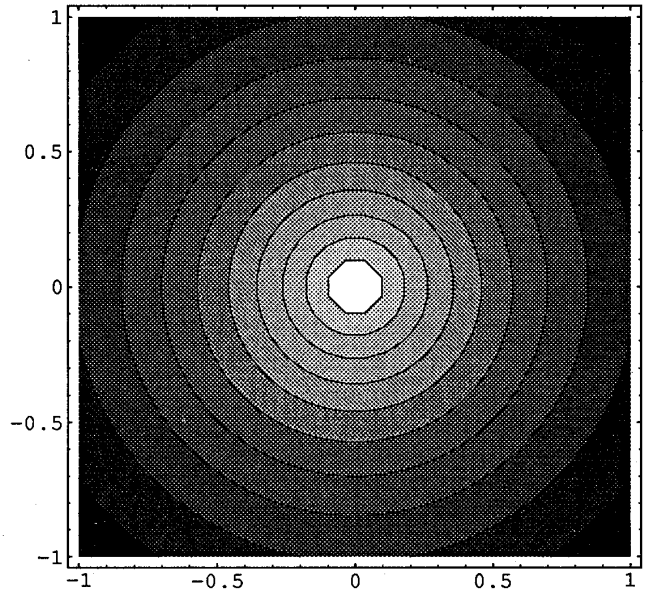
```
-SurfaceGraphics-
```

We can represent information in contour plots or density plots

In the following density plot, we have increased the nmber of plot points to improve the appearince of the resulting graph.

In[38]:=

```
ContourPlot[Exp[-Sqrt[x^2+y^2]],{x,-1,1},{y,-1,1},
            PlotPoints->30]
```



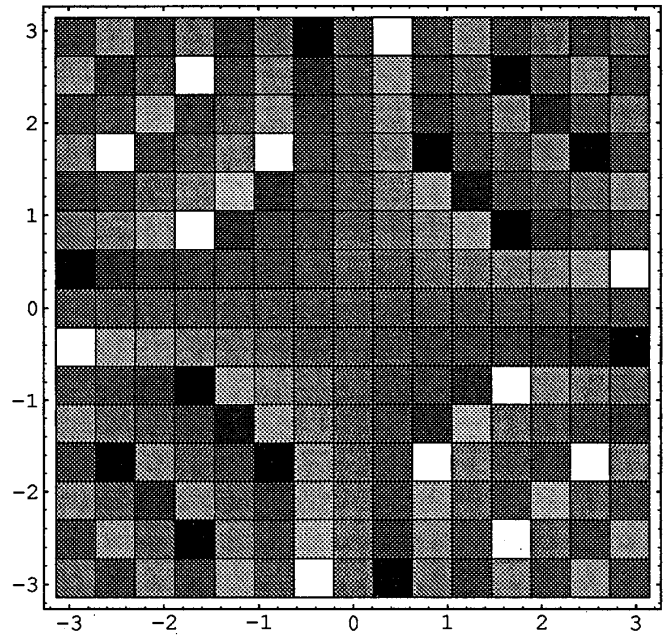
Out[38]=

```
-ContourGraphics-
```

A density plot divides the plane into an array of points and shows the value of the function at those points. Lighter regions are higher in the z direction (up out of the page).

In[39]:=

```
DensityPlot[ Tan[x y],{x,-Pi,Pi},{y,-Pi,Pi}]
```



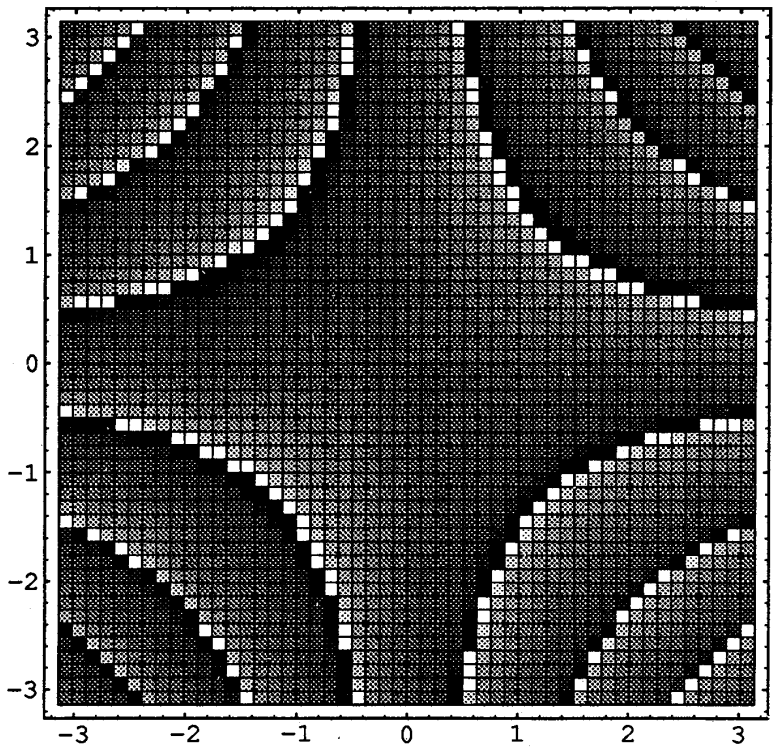
Out[39]=

```
-DensityGraphics-
```

We can increase the fineness of the plot by increasing the number of plot points

In[40]:=

```
DensityPlot[ Tan[x y],{x,-Pi,Pi},{y,-Pi,Pi},
             PlotPoints->50]
```



Out[40]=

```
-DensityGraphics-
```

□ Data Plotting

Often, we do not have an equation in closed form to display. What we have is a set of data points. In *Mathematica* terms this is a *list*. For many of the fuction plotting commands, there is a list plotting equivelant.

ListPlot
ListPlot3D
ListContourPlot
ListDensityPlot
ListPlay

Mathematica has a powerful set of commands to import data from outside files. But, we will not go into that here. we will use the Table command to generate a sample list of data points.

In[41]:=

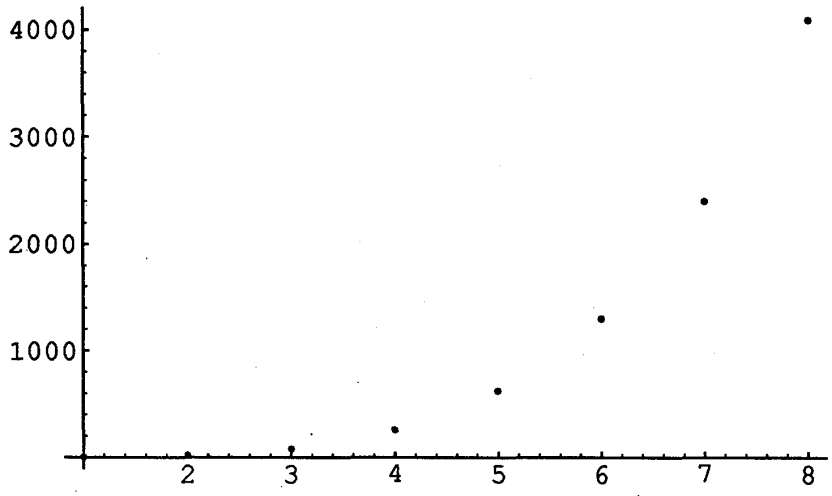
```
data = Table[{i,i^4},{i,8}]
```

Out[41]=

```
{{1, 1}, {2, 16}, {3, 81}, {4, 256}, {5, 625},
  {6, 1296}, {7, 2401}, {8, 4096}}
```

In[42]:=

```
ListPlot[ data ]
```



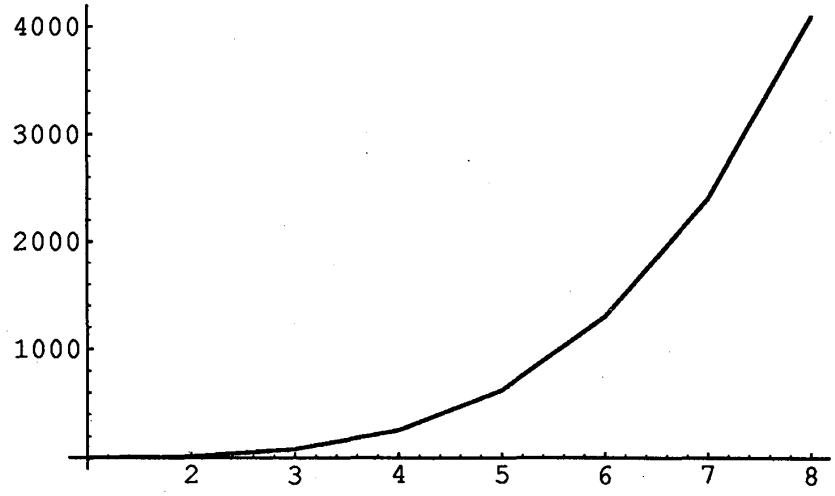
Out[42]=

```
-Graphics-
```

One of the options for ListPlot is PlotJoined. If true, it connects the data points

In[43]:=

```
ListPlot[ data, PlotJoined->True]
```



Out[43]=

```
-Graphics-
```

□ GraphicsArray

One of the features of version 2.0 is the command GraphicsArray. With it, we can make an array of plots.

In this example, we set the display function to the identity : This causes the resulting plots, **not** to be displayed. The value of j steps from 0 to Pi in step sizes of Pi/2. That is, j takes the vales 0, Pi/2, and Pi

In[44]:=

```
plotlist = Table[      Plot[ Sin[ i x + j], {x,0,2Pi},
                           DisplayFunction->Identity],
                   {i,4},{j,0,Pi,Pi/2}]
```

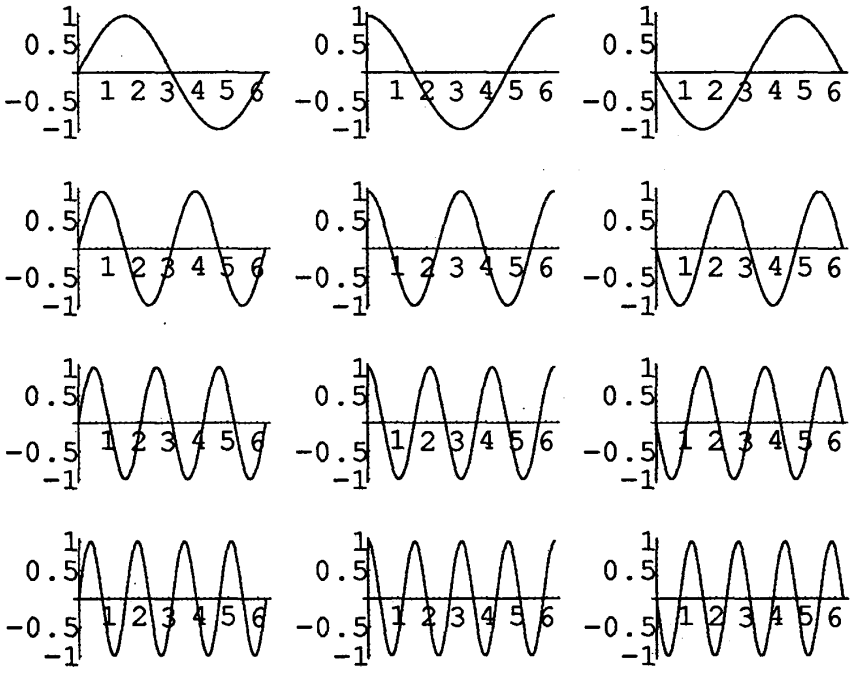
Out[44]=

```
{{-Graphics-, -Graphics-, -Graphics-},
  {-Graphics-, -Graphics-, -Graphics-},
  {-Graphics-, -Graphics-, -Graphics-},
  {-Graphics-, -Graphics-, -Graphics-}}
```

The output of the above is twelve graphics objeots. They were not displayed because the display function was set to the identity, suppressing any display. Now, we will display them as a graphics array.

In[45]:=

```
Show[ GraphicsArray[ plotlist] ]
```



Out[45]=

```
-GraphicsArray-
```

Redisplaying and Modifying Graphics

□ Show command

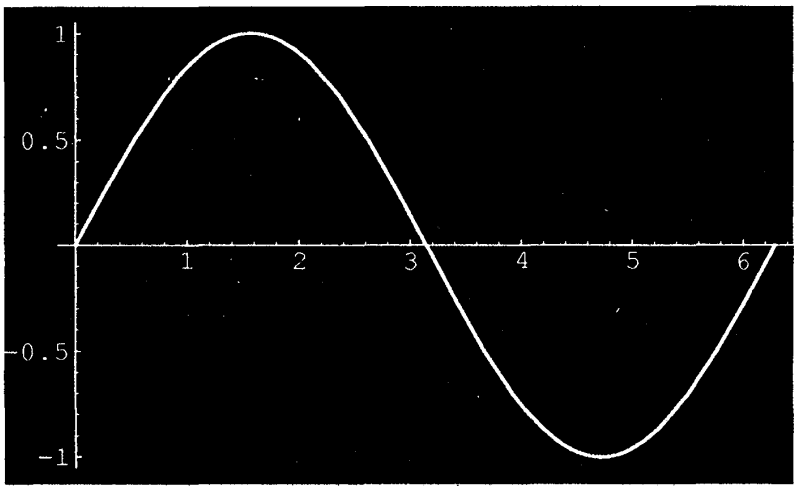
In the last section, the Show command was previewed. We use this command to redisplay, modify, or combine graphic images.

```
Show [image]
Show [image, option → setting]
Show [image1, image2]
```

Suppose we assign the plot of the sine function to the name 'plotl', but we want the background to be black.

In[46]:=

```
plot1 = Plot[ Sin[t],{t,0,2Pi},
                    Background->GrayLevel[0]]
```



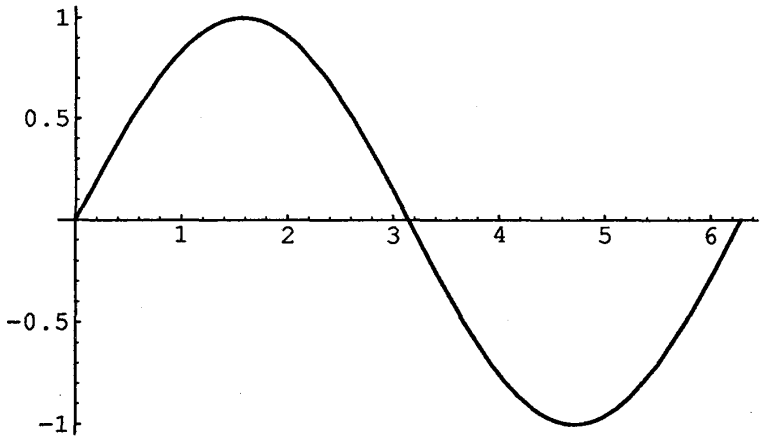
Out[46]=

```
-Graphics-
```

We can cause the plot to be shown again with the Show command. This time, we will change the background color.

In[47]:=

```
Show[plot1, Background->GrayLevel[1]]
```



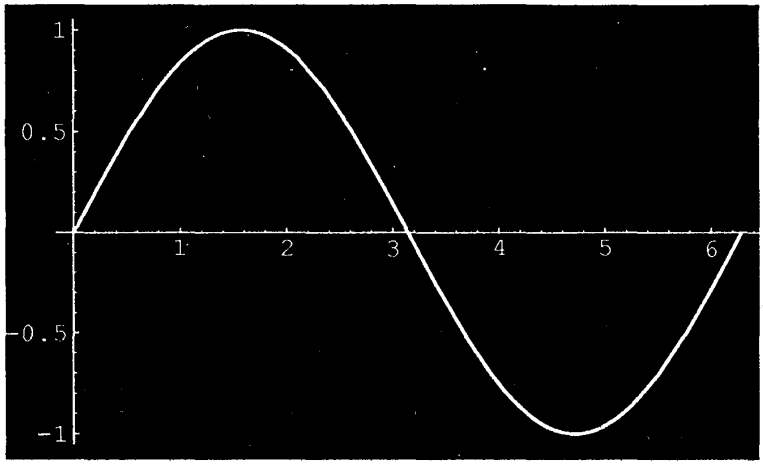
Out[47]=

```
-Graphics-
```

But, if we ask for plotl to be displayed again, the background is still the default (GrayLevel [0] , i.e. black).

In[48]:=

```
Show[plot1]
```



Out[48]=

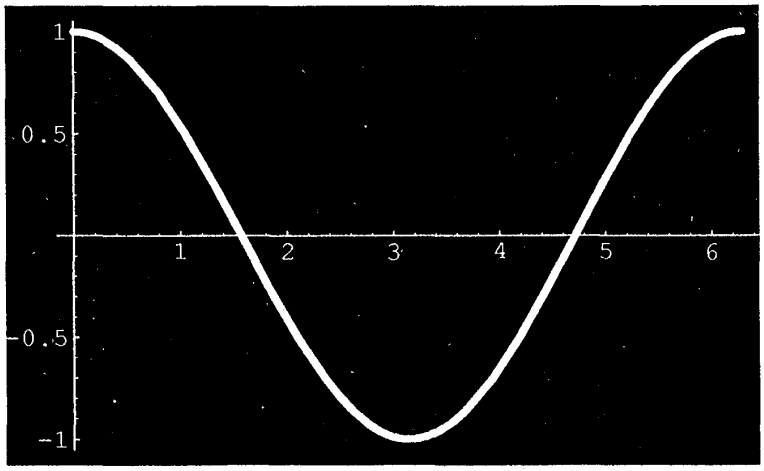
```
-Graphics-
```

Let's make plot2. We will use the Plotstyle option to make the line thicker.

The argument to the thickness command, '.01', means the thickness of the line is 1% of the size of the displayed plot. That is rather thick. (A thickness of 0.1 would be very thick, 10% of the overall size of the graphic!)

In[49]:=

```
plot2 = Plot[ Cos[t],{t,0,2Pi},
                    Background->GrayLevel[0],
                    PlotStyle->Thickness[.01]
              ]
```



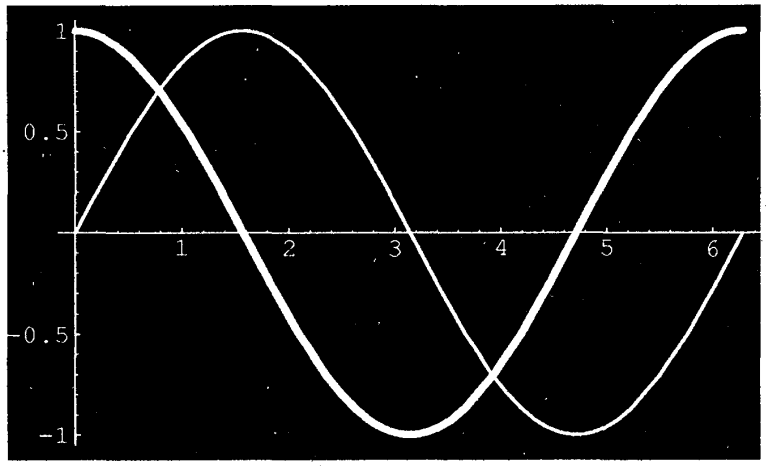
Out[49]=

-Graphics-

We can combine these two plots with the show command.

In[50]:=

```
Show[plot1,plot2]
```



Out[50]=

-Graphics-

□ Options Settings

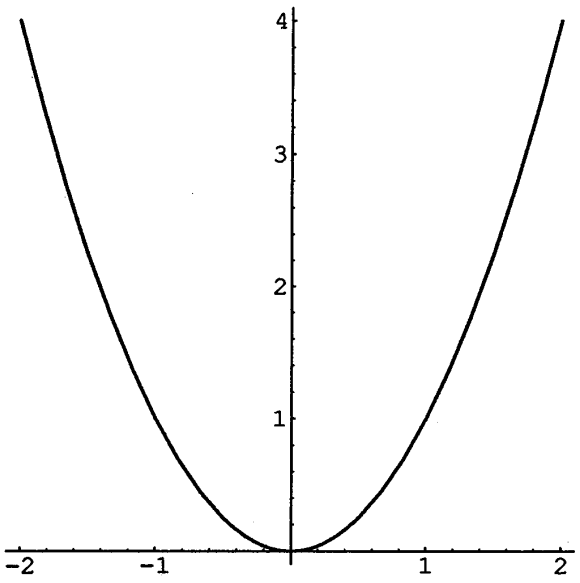
The settings allowed for different options is different depending on the function of the option. Some options are true or false, and take Boolean values. Other options take numbers, e.g. to control the number of plot points. Other options take special values.

Common settings for options are: **Automatic** (use a *Mathematica* algorithm to decide), **All** (include everything), **None** (do not include), **True**, and **False**.

The use of the Options command was covered in Preliminaries, above.

To get more detailed information, use the command Fulloptions. If we plot x squared with an automatic aspect ratio,

```
In[51]:=
Plot[x^2,{x,-2,2},AspectRatio->Automatic]
```




```
Out[51]=
-Graphics-
```

and, if we want to know what aspect ratio *Mathematica* actually chose:

```
In[52]:=
FullOptions[%,AspectRatio]

Out[52]=
1.
```

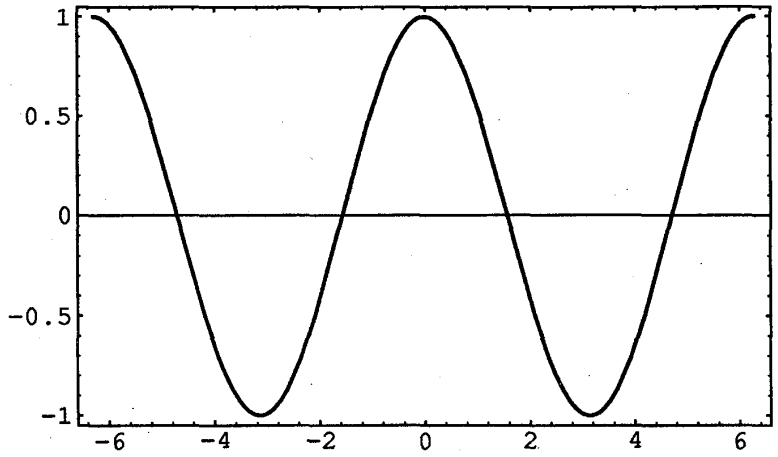
■ Options and Two-Dimesional Graphics

Here are some examples of using options to modify two-dimensional graphics.

These are only a few of more than a multitude of possible example.

In[53]:=

```
Plot[ Cos[t],{t,-2Pi,2Pi},
        Axes->{True,False},
        Frame->True
    ]
```

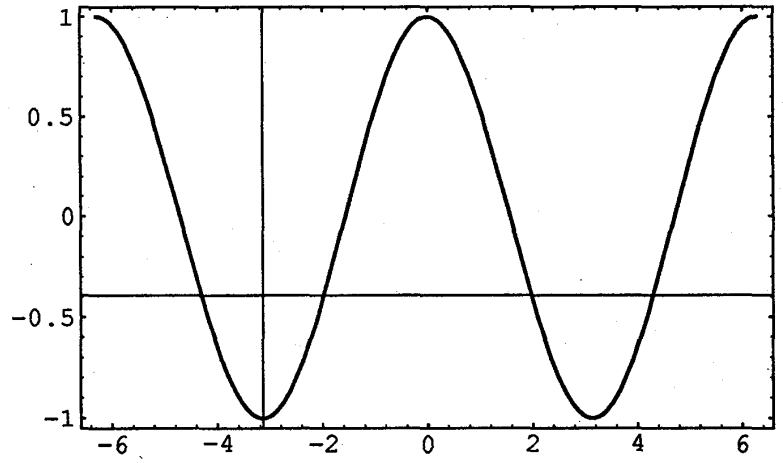


Out[53]=

```
-Graphics-
```

In[54]:=

```
Plot[ Cos[t],{t,-2Pi,2Pi},
        AxesOrigin->{-Pi,-Pi/8},
        Frame->True
    ]
```



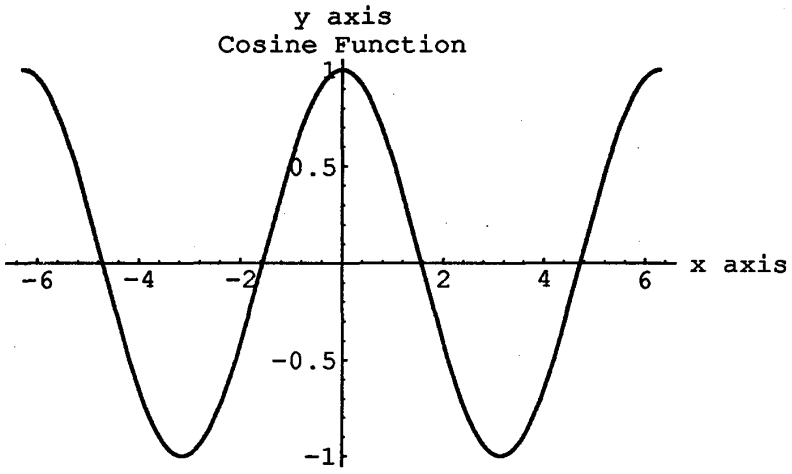
Out[54]=

```
-Graphics-
```

Labels can be added to a plot

In[55]:=

```
Plot[Cos[t],{t,-2Pi,2Pi},
        AxesLabel->{"x axis","y axis"},
        PlotLabel->"Cosine Function"
    ]
```



Out[55]=

```
-Graphics-
```

The PlotRange Option can be used to control the region which is actually shown in the plot display.

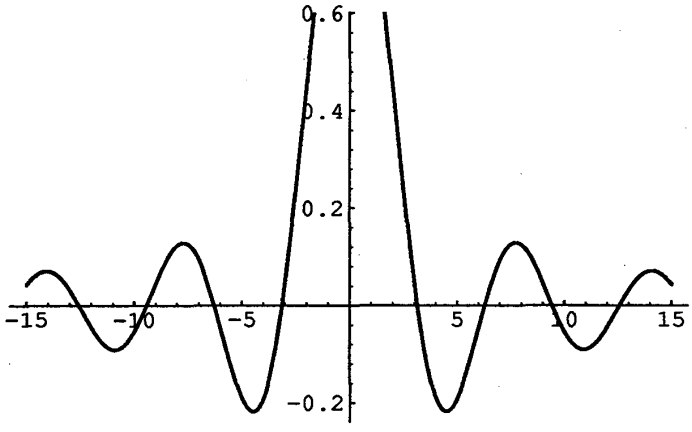
In[56]:=

```
Plot[ Sin[x]/x, {x,-15,15}]
```

```
                                        1
Power::infy: Infinite expression ---------- encountered.
                                        0.

Infinity::indet:
   Indeterminate expression 0. ComplexInfinity
     encountered.

Plot::plnr:
   CompiledFunction[{x}, <<1>>, -<<11>>e-][x]
     is not a machine-size real number at x = 0..
```



Out[56]=

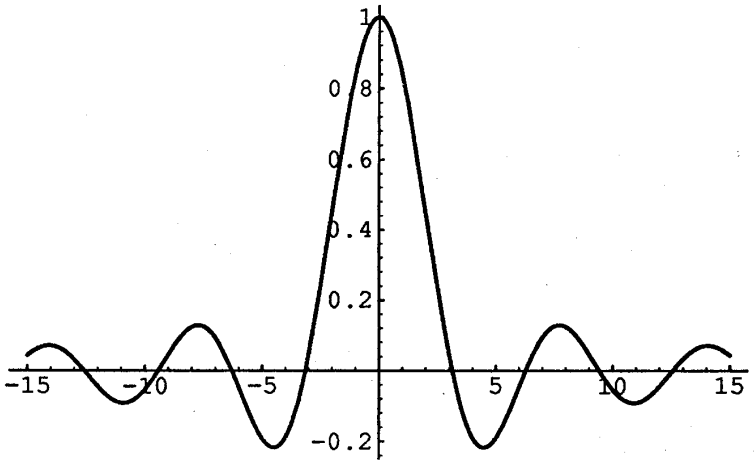
```
-Graphics-
```

Mathematica complains about the singularity at the origin, but it produces the requested plot. However, it does not show the entire range of the y-axis.

We can force this with the PlotRange option

In[57]:=

```
Show[%,PlotRange->All]
```



Out[57]=

```
-Graphics-
```

FullOptions can be used to show the value of PlotRange which was atually used.

In[58]:=

```
FullOptions[%, PlotRange]
```

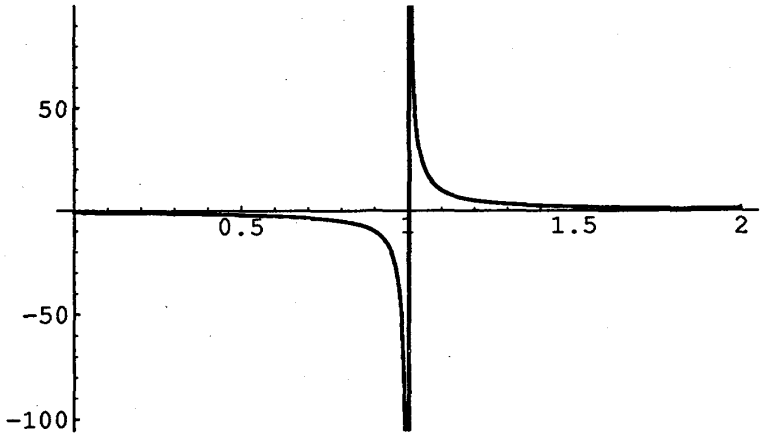
Out[58]=

```
{{-15.75, 15.75}, {-0.247658, 1.03017}}
```

We can control the range of a plot by giving specific values to the PlotRange Option. For example, if we plot a function with a singularity, we may get a larger range than we would wish.

In[59]:=

```
Plot[1/(x-1),{x,0,2}]
```



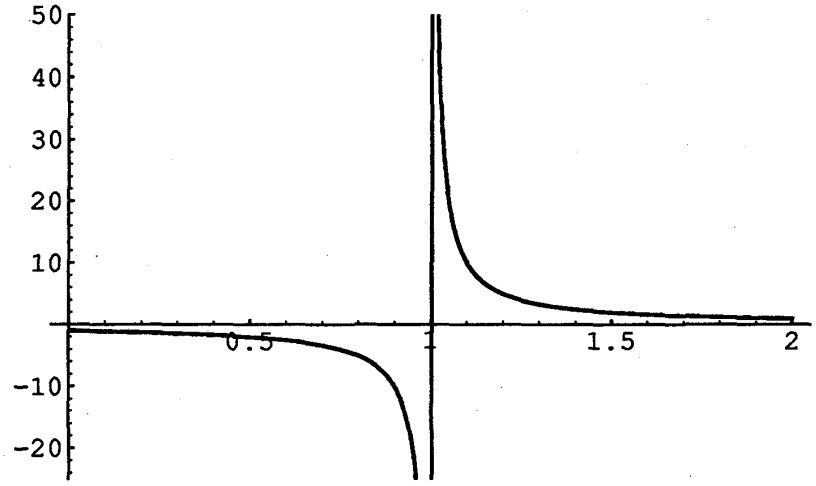
Out[59]=

```
-Graphics-
```

This can be restricted

In[60]:=

```
Show[%, PlotRange->{-25,50}]
```



Out[60]=

```
-Graphics-
```

Abelson, Harold and Andrea DiSessa. *Turtle Geometry*. MIT Press Series in Artificial Intelligeince, ed. MIT Press. Cambridge, MA : MIT Press, 1980.

Andrade, E.N.Da C. "Isaac Newton." In *World of Mathematics*, ed. James Newman. 255-276.1.NY : Simon and Schuster, 1956.

Barker, Keith, David L. Soldan, and Gordon E. Stokes. "Laboratory Experiences in Computer Science and Engineering." Computer Science Education 1 (1 1988) : 1-10.

Barwise, Jon and John Etchemendy. "Visual Information and Valid Reasoning." In Visualization in Teaching and Learning Mathematics, ed. Walter Zimmerman and Steve Cunningham. 9-24. 19.MAA, 1991.

Baxter, Nancy, Ed Dubinsky, and Gary Levin. Learning Discrete Mathematics with ISETL. NY : Springer-Verlag, 1989.

Boyer, Carl B. The History of the Calculus and its Conceptual Development. Dover, 1949.

Burton, Richard R. and John Seely Brown. "An Investigation of Computer Coaching for Informal Learning Activities." In *Intelligent Tutoring Systems*, ed. D. Sleeman and .S. Brown. London: AP, 1982.

Burton, David M. *The History of Mathematics : An Introduction.* 2 ed., 1985.

Choate, Jonathan. "Chaos and Fractal Software 〔software review〕 ."
The College Mathematics Journal, January 1991, 6569.

Clark, Ronald W. *Einstein : The Life and Times*. Avon, 1971.

Coffey, Shannon, Andre Deprit, Etienne Deprit, and Liam Healy.
"Painting the Phase Space Portrait of an Integrable Dynamical System." Science 247 (1990) : 833-836.

Cournot, Augustin. "Mathematics of Value and Demand." In *World of Mathematics*, ed. James R. Newman. 1203-1216.2. NY : Simon and Schuster, 1956.

Danby, J.M.A. "Computer Applications in Differential Equations." In *Computers and Mathematics*, ed. David A. Smith. 73-78.9 of MAA Notes Series. MAA, 1988.

Davis, Phillip J. and James A. Anderson. "Nonanalytic Aspects of Mathematics and Their Implication for Research and Education." SIAM Review 21 (1979) : 112-117.

Davis, Phillip J. and Reuben Hersh. *The Mathematical Experience* (section on Polya). 1981.

Davis, Phillip J. and Reuben Hersh. *The Mathematical Experience* (section on scientific computing). 1981.

Davis, Phillip J. and Reuben Hersh. *Descartes Dream*. Boston : Houghton Mifflin, 1986.

Devlin, Keith. "Why Computer and Mathematics." Notices of the AMS, MArch 1991, 190-191.

Drysdale, R.L. Scot, Henry F. Korth, and Allen B. Tucker. "Computer Science in Liberal Arts Colleges." Computer Science Education 1 (1 1988) : 1-35.

Dubinsky, ED. "ISETL is distributed free." 1992.

Eisenberg, Theodore and Tommy Dreyfus. "On the reluctance to visualize in Mathematics." In *Visualization in Teaching and Learning Mathematics*, ed. Walter Zimmerman and Steve Cunnningham. 25-37.19.MAA, 1991.

Gleick, James. *Chaos*. Y : Viking Penguin, 1987.

Goldenburg, E. Paul. "The Difference Between Graphing Software and Educational Graphing Software." In *Visualization in Teaching and Learning Mathematics*, ed. Walter Zim-

merman and Steve Cunnningham. 77-86.19.MAA, 1991.

Gordon, Florence S. "Computer Use in Teaching Statistics." In *Computers and Mathematics*, ed. David A. Smith. 79-83.MAA, 1988.

Hall, Leon and Stan Wagon. "Roads and Wheels." Mathematica Conference, 1992, 1992.

Hallett, Deborah Hughes. "Visualization and Calculus Reform." *In Visualization in Teaching and Learning Mathematics*, ed. Walter Zimmerman and Steve Cunnningham. 121-126.19.MAA, 1991.

Halmos, Paul. "Is Computer Teaching Harmful?" Notices of the AMS, May/June 1991, 420-423.

Hardy, G.H. "A Mathematician's Apology." In *World of Mathematics*, ed. James R. Newman. 2027-2038.4. NY : Simon and Schuster, 1956.

Hix, Deborah. "Teaching a Course in Human-Computer Interaction." Computer Science Education 1 (3 1990) : 253-268.

Howerton, Charles P. "The Impact of Pre-College computer Exposure on Student Achievement in Introductory Computer Programming Courses." Computer Science Education 1 (1 1988) : 73-84.

Kaplan, Ken. "Real Time Goes Home." BYTE, August 1992, 195-200.

Keynes, John Maynard. "Newton, the Man." In *World of Mahematics*, ed. James Newman. 277-285.1. NY : Simon and Schuster, 1956.

Klotz, Eugene A. "Visualization in Geometry : A Case Study of a Multimedia Matematics Education Project." In *Visualization in Teaching and Learning Mathematics*, ed. Walter Zimmerman and Steve Cunnningham. 95-104.19.MAA, 1991.

Leinbach, L. Carl. "The Machine in the Garden: Calculus with Computing." In *Computers and Mathematics*, ed. David A. Smith. 19-28. MAA, 1988.

Mandelbrot, Benoit B. *The Fractal Geometry of Nature*. W.H. Freeman, 1983.

McCormick, Bruce H., Thomas A. DeFanti, and Maxine D. Brown. "Visualization in Scientific Computing." Computer Graphics 21 (1987):

Miller, G.A. "The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information." The Psychological Review (1956):

Minor, Lee H. "Visualization of Limits and Limits of Visualization." The College Mathematics Journal 23 (1 1992): 48-51.

Newman, James R., ed. *World of Mathematics*. NY: Simon and Schuster, 1956.

NRC, National Research Council. "Everybody Counts: A Report to the Nation About the Future of Mathematics Education." National Research Council, 1989.

NRC, National Research Council. "Renewing U.S. Mathematics." National Research Council, 1990.

Orzech, Morris. "Using Computers in Teaching Linear Algebra." In Computers and Mathematics, ed. David A. Smith. 63-67.MYY, 1988.

Papert, Seymour. *Mindstorms*. NY: Basic Books, 1980.

Parker, Jeff, Robert Cupper, Charles Kelemen, Dick Molnar, and Greg Scragg. "Laboratories in the Computer Science Curriculum." Computer Science Education 1 (3 1990): 205-221.

Poincaré, Henri. "Mathematical Creation." In *World of Mahematics*, ed. James Newman. 2041-2050.4. NY: Simon and Schuster, 1956.

Rival, Ivan. "Picture Puzzling: Mathematicians are Rediscovering the Power of Pictorial Reasoning." The Sciences 27 (1987): 41-46.

Sambursky, S. *The Physical World of the Greeks*. Translated by Metron Dagut. Princeton, NJ: Princeton U. Press, 1956.

Sleeman, D. and J.S.Brown, ed. *Intelligent Tutoring Systems*. Computers and People. London: Academic Press, 1982.

Snell, J.Laurie and John Finn. "The Use of the Computer in a Probability Course." In *Computers and Mathematics*, ed. David A. Smith. 85-96.MAA, 1988.

Steen, Lynn A. "The Science of Patterns." Science 29 (1988):

Stoutemyer, David R. "Crimes and Misdemeanors in the Compter Algebra Trade." Notices of the AMS, September 1991, 778-785.

Tall, David. "Recent Developments in the Use of Computers to Visualize and Symbolize Calculus Concepts." In *The Laboratory Approach to Teaching Calculus*, ed. L. Carl Leinbach. 15-25.20 of MAA Notes Series. MAA, 1991.

Tucker, Thomas W., ed. Priming the Calulus Pump: Innovations and Resources. MAA Notes. Mathematical Association of America, 1991.

Tukey, John. *Exploratory Data Analysis*. Behavioral Science: Quantitive Methods, ed. Fredeick Mosteller. NY: Addson-Wesley, 1976.

Varian, Hal R. *Symbolic Optimization*. University of Michigan, 1992.

Wagon, Stan. *Mathematica in Action*. NY: Freeman, 1991.

Wei, Sha Xin. "Mathematica 2.0 (review)." Notices of the AMS, May/June 1992, 428-435.

Weissglass, Julian and Deborah Cummings. "Dynamic Visual Experiments with Random Phenomena." In MAA Notes, ed. Warren Page. 215-223. Mathematical Association of America, 1991.

Zimmerman, Walter and Steve Cunnningham. "Editor's Introduction : What is Mathematical Visualization." In *Visualization in Teaching and Learning Mathematics*, ed. Walter Zimmerman and Steve Cunnningham. 1-8.19.MAA, 1991.

Zimmerman, Walter. "Visualization Thinking in Calculus." In MAA notes, ed. Warren Page. 127-137. Mathematical Association of America, 1991.